WHAT IS CLAIMED IS:

1. A method of processing a structured text comprising the steps of:

creating, from the structured text, a tokenizer text including simplex constituents constructed in accordance with a predetermined set of tokenized rules of a token pattern knowledge base, each tokenizer rule defining a simplex constituent;

creating, from the tokenized text, a parsed text including complex constituents constructed in accordance with a predetermined set of parser rules of a parser rule knowledge base, each parser rule defining a complex constituent; and

creating, from the parsed text, a processed text including message elements constructed in accordance with a predetermined set of interpreter rules of an interpretation knowledge base, each interpreter rule defining a message element.

2. The method of claim 1, wherein the processed text identifies and provides an interpretation of the message elements of the structured text for text-to-speech synthesis.

3. The method of claim 1, wherein the step of creating the tokenized text comprises the steps of:

providing a simplex constituent buffer to store the simplex constituents; and

processing a line of text in the structured text, resulting in a line of tokenized text including at least one token, until all lines of text have been processed,

wherein the resulting tokenized text includes the tokens and simplex constituents constructed in accordance with the predetermined tokenizer rules, and

wherein each simplex constituent has a start marker applied to a start token of the simplex constituent and an end marker applied to an end token of the simplex constituent.

4. The method of claim 3, wherein the step of processing the line of text comprises the steps of:

processing the line of text as one token if the line of text matches a line pattern in the token pattern knowledge base, to result in a matched line pattern; and

processing the line of text as at least one word if the line of text fails to match any line pattern in the token pattern knowledge base.

5. The method of claim 4, wherein the step of processing the line of text as one token includes the steps of:

creating a line-spanning token, which includes the line of text, wherein the start marker and the end marker identify the simplex constituent corresponding to the matched line pattern in the token pattern knowledge base;

creating a full-line token simplex constituent which spans the line-spanning token; and

storing the full-line token simplex constituent in the simplex constituent buffer.

6. The method of claim 4, wherein the step of processing the first word token includes the steps of:

creating a current token for each word in the line;

processing a first word token which matches a start line keyword pattern in the token pattern knowledge base, to result in a matched start line keyword, a full-line simplex constituent and a stored end marker;

processing a word which matches a word pattern in the token pattern knowledge base, to result in a matched word pattern; and

finalizing the stored end marker.

7. The method of claim 6, wherein the step of processing the first word token includes the steps of:

assigning the start marker to the current token, wherein the start marker identifies the simplex constituent corresponding to the matched start line keyword in the token pattern knowledge base;

creating the full-line simplex constituent corresponding to the matched start line keyword in the token pattern knowledge base;

adding the current token to the full-line simplex constituent as a start token of the full-line simplex constituent; and

saving the stored end marker, wherein the stored end marker identifies the simplex constituent corresponding to the matching start line keyword in the token pattern knowledge base.

8. The method of claim 6, wherein the step of processing the word includes the steps of:

adding the start marker and the end marker to the current token, wherein the start marker and the end marker identify the simplex constituent corresponding to the matched word pattern in the token pattern knowledge base;

creating a single-word simplex constituent spanning the current token which corresponds to the matched word pattern in the token pattern knowledge base; and

adding the single-word simplex constituent to the simplex constituent buffer.

9. The method of claim 6, wherein the step of processing the word includes the steps of:

adding the stored end marker to the current token;

assigning the current token to the full-line simplex constituent; and

adding the full-line simplex constituent to the simplex constituent buffer.

10. The method of claim 1, wherein the step of creating the parsed text comprises the steps of:

providing a complex constituent buffer to store the complex constituents; and

processing the tokenized text until all possible complex constituents have been created.

11. The method of claim 10, wherein the step of processing the tokenized text comprises the steps of:

searching for a sequence of complex constituent input elements that matches one of the predetermined parser rules in the parser rule knowledge base to result in a matched complex constituent input sequence;

creating the complex constituent corresponding to the matched complex constituent input sequence;

adding a start label to a start token of the complex constituent and an end label to an end token of the complex constituent, wherein the start label and the end label identify the complex constituent corresponding to the matched complex constituent input sequence in the parser rule knowledge base; and

adding the complex constituent to the complex constituent buffer.

12. The method of claim 11, wherein the sequence of complex constituent input elements includes at least one of (a) at least one token; (b) at least one simplex constituent which spans at least one token; and (c) at least one complex constituent which spans at least one token.

13. The method of claim 1, wherein the step of creating the processed text comprises the steps of:

creating, from the parsed text, a tree structure including a root node, at least one internal node and leaves, wherein the root node dominates the internal nodes and leaves, the root node and each of the internal nodes in the tree structure have corresponding interpreter functions, and the leaves are tokens of the parsed text; and

traversing the tree structure wherein the interpreter functions associated with the root node and each internal node are executed to result in the corresponding message element.

14. The method of claim 13, wherein the interpretation function includes a default function.

15. The method of claim 14, wherein the default function includes concatenation.

16. The method of claim 13, further comprising a user-specified function to produce the message element

17. The method of claim 13, further comprising an optional post-processor directive to produce the message element.

18. The method of claim 13, further comprising the step of interpreting tags of an output using a text-to-speech synthesizer, wherein the tags correspond to at least one of (a) the start marker and the end marker in the tokenized text, and (b) the start label and the end label in the parsed text.

19. The method of claim 18, wherein the tags are SGML tags.

20. A program for processing a structured text stored on a computer readable medium comprising:

a computer readable program code for creating a tokenized text, from the structured text, including simplex constituents constructed in accordance with a predetermined set of tokenizer rules of a token pattern knowledge base;

a computer readable program code for creating a parsed text, from the tokenized text, including complex constituents constructed in accordance with a predetermined set of parser rules of a parser rule knowledge base; and

a computer readable program code for creating a processed text, from the parsed text, including message elements constructed in accordance with a predetermined set of interpreter rules of an interpretation knowledge base.